Learning Conformation Rules

AUTHOR(S):

Maruyama, Osamu; Furuichi, Emiko; Kuhara, Satoru; Miyano, Satoru

CITATION:

Maruyama, Osamu ...[et al]. Learning Conformation Rules. 数理解析研究所講究録 1997, 992: 28-35

ISSUE DATE:

1997-05

URL:

http://hdl.handle.net/2433/61160

数理解析研究所講究録
992巻 1997年 28-35

Learning Conformation Rules

Osamu Maruyama (丸山 修)† Emiko Furuichi (古市 恵美子)‡
Satoru Kuhara (久原 哲)§ Satoru Miyano (宮野 悟)†

{maruyama,miyano}@ims.u-tokyo.ac.jp
{emiko,kuhara}@grt.kyushu-u.ac.jp

† Human Genome Center, University of Tokyo
‡ Fukuoka Women's Junior College
§ Graduate School of Genetic Resources Technology, Kyushu University

要旨

We define a hypergraph representation of a protein which captures its tertiary structure in a loose way. By using the notions of hypergraphs, we define a conformation rule as a kind of hypergraph rewriting. With a conformation rule, a procedure of conformation from sequences is described. Then we discuss a method of learning a conformation rule from a collection of hypergraph representations of proteins. A polynomial-time PAC-learning algorithm is shown for a class of conformations. This algorithm is now being implemented with Common Lisp for experiments.

1 Introduction

Protein conformation has been analyzed in terms of free energy. Various computational methods have been extensively developed for searching minimal energy conformations. For example, a recursive method is developed to identify a large number of low energy conformations [11] and genetic algorithms are also applied to this problem [8, 12]. Another interesting heuristic method is the hydrophobic zipper method by [2]. Based on the fact many hydrophobic contacts are topologically local, the hydrophobic zipper method randomly selects hydrophobic contacts among neighbors in a sequence and zips up other hydrophobic contacts.

Inspired by this hydrophobic zipper method, but apart from the free-energy minimization problem, we define a conformation rule as a rewriting rule of hypergraphs [1]. For this definition, we introduce a hypergraph representation of a protein by which the three dimensional structure of the protein is loosely captured. A conformation rule is applied to a sequence from local toward global as in the hydrophobic zipper method, and finally produces a hypergraph which represents the structure.

We then consider the problem of learning conformation rules from hypergraph representations of proteins. A conformation is defined as a function from sequences to hypergraphs. Thus the problem is to learn functions from examples. The PAC-learning paradigm was extended to include functions by Natarajan and Tadepalli [7] and some results on concept learning have been extended for functions [5, 6]. This paper has two contributions. One is a formulation of conformation rules by using hypergraphs, and the other is a polynomial-time PAC-learning algorithm for a class which is defined by this new concept of conformation rules. We are now implementing this algorithm with Common Lisp for experiments by using 153 proteins in PDB whose tertiary structures are already determined.

2 Preliminaries

For an undirected graph $G = (V, E)$, we denote by $d(u, v)$ the length of the shortest path between u and v. The *diameter* $\delta(G)$ is defined to be $\max\{d(u, v) \mid u, v \in V\}$. If G is not connected, $\delta(G) = \infty$.

For an undirected graph $G = (V, E)$, we say that $(i_1, \ldots, i_k)$ is a *k-cycle* in G if $i_1, \ldots, i_k$ are mutually distinct nodes of G and $\{i_1, i_2\}, \ldots, \{i_{k-1}, i_k\}, \{i_k, i_1\}$ are edges in E.

We denote the cardinality of a set S by $|S|$. For an alphabet Ω, Ω^+ denotes the set of all nonempty strings over Ω. The length of a string $x \in \Omega^+$ is denoted by $|x|$. For $n \geq 1$, $\Omega^{[n]} = \{x \in \Omega^+ \mid |x| \leq n\}$.

2.1 Hypergraphs

A *hypergraph* $H = (V, F)$ consists of a set V of nodes and a set F of *hyperedges*, each of which is a nonempty subset of V [1]. In this paper we assume that $|e| \geq 2$ for all $e \in F$ without any notice. A *chain-hypergraph* is a hypergraph $H = (V, F)$ such that $V = \{1, 2, \ldots, n\}$ for some $n \geq 1$ and each $\{i, i+1\}$ is contained in some hyperedge in F for $1 \leq i \leq n-1$, i.e., there is e in F with $\{i, i+1\} \subseteq e$.

The *rank* of H is $r(H) = \max_{e \in H} |e|$. For a node v, the *degree of* v is $d_H(v) = |\{e \mid e \in F,\ v \in e\}|$ and the *degree of* H is $d(H) = \max_{v \in V} d_H(v)$. For a node v of H, the set of hyperedges containing v is called the *neighborhood* of v and is denoted by $N_H(v) = \{e \mid e \in F,\ v \in e\}$.

For a set F of hyperedges, we call

$$\text{simplify}(F) = F - \{e \in F \mid \text{there is } e' \text{ in } F \text{ with } e \subseteq e' \text{ and } e \neq e'\}$$

the *simplification* of F. We say that $H = (V, F)$ is *simple* if $F = \text{simplify}(F)$,i.e., there are no two distinct hyperedges e_1 and e_2 in F satisfying $e_1 \subseteq e_2$.

In this paper we consider a hypergraph $H = (V, F)$ whose nodes are labeled with a mapping $\varphi : V \to \Delta$, where Δ is an alphabet. It is denoted by $H = (V, F, \varphi)$ and called a hypergraph over Δ. We confuse $H = (V, F, \varphi)$ with $H = (V, F)$ without any notice.

2.2 PAC-learnability of a class of functions

This section reviews some notions and results on the PAC-learnability of a class of functions by following Natarajan [6].

In this section, the alphabet Ω is assumed to be finite.

Definition 1. [5] Let $\mathcal{G}$ be a class of functions from a finite set X to a finite set Y. The *generalized VC-dimension* of $\mathcal{G}$, denoted by $D(\mathcal{G})$, is the maximum over the sizes $|Z|$ of subsets $Z \subseteq X$ such that there exist two functions f and g in $\mathcal{G}$ satisfying the following conditions:

1. $f(x) \neq g(x)$ for all $x \in Z$.
2. For all $Z_1 \subseteq Z$, there exists $h \in \mathcal{G}$ that agrees with f on Z_1 and with g on $Z - Z_1$.

Lemma 1. [5] Let $\mathcal{G}$ be a class of functions from a finite set X to a finite set Y. Then

$$2^{D(\mathcal{G})} \leq |\mathcal{G}| \leq |X|^{D(\mathcal{G})}|Y|^{2 \cdot D(\mathcal{G})}.$$

□

Let $f : \Omega^+ \to \Omega^+$. For integers $n_1, n_2 \geq 1$, the projection $f^{[n_1][n_2]}$ of f on $\Omega^{[n_1]} \times \Omega^{[n_2]}$ is the function $f^{[n_1][n_2]} : \Omega^{[n_1]} \to \Omega^{[n_2]}$ defined by $f^{[n_1][n_2]}(x) = f(x)$ if $f(x)$ is in $\Omega^{[n_2]}$ for all x in $\Omega^{[n_1]}$. If there is some x in $\Omega^{[n_1]}$ such that $f(x)$ is not in $\Omega^{[n_2]}$, then $f^{[n_1][n_2]}$ is undefined. For a class $\mathcal{F}$ of functions from Ω^+ to Ω^+, we define $\mathcal{F}^{[n_1][n_2]} = \{f^{[n_1][n_2]} \mid f \in \mathcal{F},\ f^{[n_1][n_2]} \text{ is defined}\}$.

Definition 2. Let $\mathcal{F}$ be a class of functions from Ω^+ to Ω^+ with a representation R. An algorithm Q is said to be a *polynomial-time fitting* for $\mathcal{F}$ in representation R if the following conditions hold:

1. Q is a polynomial-time algorithm taking as input a finite subset S of $\Omega^+ \times \Omega^+$.
2. If there exists a function in $\mathcal{F}$ that is consistent with S, Q outputs a name of the function in representation R.

We say that $\mathcal{F}$ is of *polynomial-dimension* if there is a polynomial $p(n_1, n_2)$ in n_1 and n_2 such that $D(\mathcal{F}^{[n_1][n_2]}) \leq p(n_1, n_2)$.

We say that $\mathcal{F}$ is of *polynomial-expansion* if there exists a polynomial $q(n)$ such that for all $f \in \mathcal{F}$ and $x \in \Omega^+$, $|f(x)| \leq q(|x|)$.

The following theorem will be used to prove a result in Section 5 on the PAC-learnability of conformation rules. We do not provide a formal definition of the PAC-learnability of a class of functions. The readers are referred to [5] or Chapter 5 in [6].

Theorem 1. [5] Let $\mathcal{F}$ be a class of functions from Ω^+ to Ω^+ with a representation R. $\mathcal{F}$ is polynomial-time PAC-learnable in R if the following hold:

1. $\mathcal{F}$ is of polynomial-dimension.
2. $\mathcal{F}$ is of polynomial-expansion.
3. There exists a polynomial-time fitting for $\mathcal{F}$ in R. □

3 Hypergraph Representation of a Protein

Let P be a protein of the primary structure $A_1A_2\cdots A_n$. Its tertiary structure is usually represented by a sequence of the positions of amino acid residues in the three dimensional space as $(p_1, A_1), (p_2, A_2), \ldots, (p_n, A_n)$, where $p_i = (x_i, y_i, z_i)$ is the position of the amino acid residue A_i for $1 \leq i \leq n$. The distance between p_i and p_j is denoted by $|p_i - p_j|$. Let Σ be the alphabet consisting of symbols representing the amino acid residues.

Let $\varepsilon > 0$ be a real number. For a protein P with a tertiary structure (p_1, A_1), (p_2, A_2), $\ldots, (p_n, A_n)$, let $G_P = (V, E)$ be an undirected graph defined as follows:

1. $V = \{1, 2, \ldots, n\}$.
2. For any $i = 1, \ldots, n-1$, $\{i, i+1\}$ is in E.
3. For any distinct i, j in V with $|p_i - p_j| \leq \varepsilon$, $\{i, j\}$ is in E.

We call the undirected graph $G_P = (V, E)$ the *structure graph* of P with ε-range.

For a positive integer k, let $\tilde{F}_1 = \{e \mid e \subseteq V,\ 2 \leq |e| \leq k,\ G_P[e] \text{ is a complete graph}\}$, where $G_P[e]$ is the node-induced subgraph of e in G_P [3]. Let $F = \text{simplify}(\tilde{F}_1)$ and let $\varphi : V \to \Sigma$ be a mapping defined by $\varphi(i) = A_i$ for $1 \leq i \leq n$. Then a hypergraph $H_P = (V, F, \varphi)$ is a simple chain-hypergraph over Σ. We call $H_P = (V, F, \varphi)$ the *k-hypergraph representation* of P by *complete graphs.*

We have another ways to define the k-hypergraph representation of P. In the same way as the above definition of H_P, by using $\tilde{F}_2 = \{\{i_1, \ldots, i_j\} \mid i_1, \ldots, i_j \in V,\ 2 \leq j \leq k, (i_1, \ldots, i_j) \text{ is } j\text{-cycle in } G_P\}$ instead of $\tilde{F}_1$, we can define the k-hypergraph representation of P by *cycles.* When we employ $\tilde{F}_3 = \{e \mid e \subseteq V,\ 2 \leq |e| \leq k,\ \delta(G_P[e]) \leq c\}$ for some constant c, we can also define the k-hypergraph representation of P by *graphs with diameter at most c.*

Instead of the explicit representation with amino acid residues, it is often used to classify the amino acid residues into several categories (e.g., [2, 9, 10]). In order to deal with such cases, we represent a protein in a more extended way. Namely, we consider simple chain-hypergraphs whose nodes are labeled with some "colors", which are not necessarily the same as the amino acid residues. Let Δ be an alphabet which consists of such "colors" labeling the nodes of hypergraphs.

In this paper, we assume that the tertiary structure of a protein is represented by a simple chain-hypergraph over some alphabet Δ in a way mentioned above.

4 Conformation Rules

In this section, we define a conformation rule which transforms strings over Δ to chain-hypergraphs over Δ. Then for a chain-hypergraph G, we obtain a simple chain-hypergraph by the simplification of G.

Let $CH(\Delta)$ be the set of all chain-hypergraphs over Δ. A *conformation* over Δ is a function $c : \Delta^+ \to CH(\Delta)$ such that $c(s) = (V, F, \psi)$ for a string $s = x_1 \cdots x_n \in \Delta^+$ satisfies $V = \{1, \ldots, n\}$ and $\psi(i) = x_i$ for $1 \leq i \leq n$.

We give a way of defining a conformation by introducing conformation rules over Δ.

Definition 3. A *bundle rule* over Δ is a pair $\rho = (B, U)$ of a hypergraph $B = (V, F, \psi)$ over Δ and a subset U of V satisfying the following conditions:

1. $|U| \geq 2$.
2. U is not in F.
3. For any hyperedge e in F, $e \cap U \neq \emptyset$.

We call $|U|$ the *bundle size* of ρ. The *degree* of ρ is defined to be $d(B)$. The *rank* of ρ is defined to be $r(B)$.

Definition 4. A *conformation unit* over Δ is a finite set $\gamma = \{(B_1, U_1), \ldots, (B_t, U_t)\}$ of bundle rules over Δ. We say that a conformation unit γ is of *rank* k (*degree* d) if $|U_i| \leq k$ and $r(B_i) \leq k$ ($d(B_i) \leq d$) for $1 \leq i \leq t$. A *conformation rule* over Δ is a sequence $(\gamma_1, \ldots, \gamma_m)$ of conformation units over Δ. We say that a conformation rule is of *rank* k (*degree* d) if each conformation unit is of rank k (degree d). We denote by Γ_Δ the set of all conformation units over Δ and by $\Gamma_{(k,d,\Delta)}$ the set of all conformation units over Δ such that the rank is at most k and the degree is at most d for integers $k \geq 2$ and $d \geq 1$.

Remark 1. Obviously, Γ_Δ is infinite. Note that $\Gamma_{(k,d,\Delta)}$ is finite if Δ is finite. On the other hand, $\bigcup_{k \geq 2} \Gamma_{(k,d,\Delta)}$ and $\bigcup_{d \geq 1} \Gamma_{(k,d,\Delta)}$ are infinite.

Definition 5. Let (B_1, U_1) and (B_2, U_2) be bundle rules over Δ with $B_1 = (V_1, F_1, \psi_1)$ and $B_2 = (V_2, F_2, \psi_2)$, respectively. We say that (B_1, U_1) is *isomorphic* to (B_2, U_2), , denoted by $(B_1, U_1) \approx (B_2, U_2)$, if there is a bijection $\iota : V_1 \to V_2$ such that (1) $\psi_1(v) = \psi_2(\iota(v))$ for all v in V_1, (2) $\iota(U_1) = U_2$, (3) $\iota(e_1) \in F_2$ for all $e_1 \in F_1$, and (4) $\iota^{-1}(e_2) \in F_1$ for all $e_2 \in F_2$.

```
Input:  a conformation rule (γ_1, ..., γ_m) over Δ of rank k and
        a string s = x_1 ··· x_n in Δ^+
Output: a hyper graph H_s = (V_s, F_s, ψ_s)
procedure CONFORM((γ_1, ..., γ_m), s)
begin
  V_s := {1, ..., n};
  let ψ_s be a mapping defined by ψ_s(i) = x_i for 1 ≤ i ≤ n;
  F := {{i, i+1} | 1 ≤ i ≤ n − 1};
  τ := min{n, m};
  for ℓ ← 1 to τ do
  begin
    w := ℓ + 2; /* w is the window size */
    TEMP := ∅;
    foreach i : 1 ≤ i ≤ n − w + 1 do
    begin
      j := i + w − 1;
      foreach e : e ⊆ {i, ..., j} with |e| ≤ k do
      begin
        F̃ := ⋃_{l∈e} N_H(l), where H = (V_s, F, ψ_s);
        Ṽ := {u | u ∈ e' for some e' ∈ F̃};
        ψ̃ := ψ_s|_Ṽ; /* the restriction of ψ_s to Ṽ */
        if B̃ = (H̃, e) ≈ B for some B in γ_ℓ, where H̃ = (Ṽ, F̃, ψ̃);
            then add a hyperedge e to TEMP;
      end;
    end;
    F := F ∪ TEMP;
  end;
  F_s := F;
end
```

図 1: Algorithm CONFORM

```
Input:  a chain-hypergraph H = (V, F, ψ) over Δ of rank k
Output: a conformation rule (γ_1, ..., γ_m) over Δ of rank k
procedure EXTRACT(H, m)
begin
  F̃ := {{i, i+1} | 1 ≤ i ≤ n − 1};
  for ℓ ← 1 to m do
  begin
    w := ℓ + 2; /* w is the window size */
    TEMP := ∅; γ_ℓ := ∅;
    foreach i : 1 ≤ i ≤ n − w + 1 do
    begin
      j := i + w − 1;
      foreach X : X ⊆ {i + 1, ..., j − 1} with |X| ≤ k − 2 do
      if there is e ∈ F with {i} ∪ X ∪ {j} ⊆ e then
        begin
          U := {i} ∪ X ∪ {j};
          F_U := ⋃_{l∈U} N_D(l), where D = (V, F̃, ψ);
          V_U := {u | u ∈ e' for some e' ∈ F_U};
          ψ_U := ψ|_{V_U}; /* the restriction of ψ to V_U */
          add (B_U, U) to γ_ℓ, where B_U = (V_U, F_U, ψ_U);
          add U to TEMP;
        end;
    end;
    F̃ := F̃ ∪ TEMP;
  end;
end
```

図 2: Algorithm EXTRACT

Let $\sigma = (\gamma_1, \ldots, \gamma_m)$ be a conformation rule over Δ. We apply the conformation rule σ to a string $s = x_1 \cdots x_n$ in Δ^+ in the following way.

We start with a hypergraph $H_1 = (V_s, F_1, \psi_s)$, where $V_s = \{1, \ldots, n\}$, $F_1 = \{\{i, i+1\} \mid 1 \leq i \leq n-1\}$ and $\psi_s(i) = x_i$ for $1 \leq i \leq n$.

We regard conformation as a process of creating new hyperedges by enlarging the "window" on V_s corresponding to the string $s = x_1 \cdots x_n$ from smaller to larger. A *window* of size w at position i is an interval $[i, \ldots, i+w-1]$ consisting of consecutive w nodes in V_s.

At stage ℓ $(1 \leq \ell \leq m)$, the window size is set to be $w = \ell + 2$. In a window $[i, \ldots, i+w-1]$ at the ℓth stage, bundle rules in the lth conformation unit γ_ℓ in σ are applied to create new hyperedges e such that e consists of only nodes in $[i, \ldots, i+w-1]$. A new creation of a hyperedge e in the window depends on the structure of the neighborhood of e in the hypergraph $H_\ell = (V_s, F_\ell, \psi_s)$. Namely, we consider a subhypergraph $\tilde{H} = (\tilde{V}, \tilde{F}, \tilde{\psi})$ such that $\tilde{F} = \{f \mid f \in F_\ell,\ f \cap e \neq \emptyset\}$, $\tilde{V} = \bigcup_{f \in \tilde{F}} f$ and $\tilde{\psi}$ is the restriction of ψ_s to $\tilde{V}$. A new hyperedge e will be created if there is a bundle rule B in γ_ℓ which is isomorphic to $(\tilde{H}, e)$.

After creating all new hyperedges at the ℓth stage, these hyperedges are added to F_ℓ. A formal description is given in Fig. 1.

The following proposition is obvious by definition:

Proposition 1. For a conformation rule $(\gamma_1, \ldots, \gamma_m)$ over Δ of rank k and a string $s = x_1 \cdots x_n$ in Δ^+, CONFORM$((\gamma_1, \ldots, \gamma_m), s)$ is a chain-hypergraph of rank k. □

Definition 6. For a conformation rule $\sigma = (\gamma_1, \ldots, \gamma_m)$ over Δ, we define a conformation c_σ by $c_\sigma(s) =$ CONFORM(σ, s) for s in Δ^+.

For a string $s = x_1 \cdots x_n$ and a conformation rule σ over Δ of rank k, let $G_s = (V_s, F_s, \psi_s)$ be the chain-hypergraph obtained by CONFORM(σ, s) and let F =simplify(F_s). Then $G = (V_s, F, \psi_s)$ is a simple chain-hypergraph.

5 PAC-Learning of Conformation

Let $\mathcal{C} = \{c_\sigma \mid \sigma \in \Gamma_\Delta^+\}$. For $n \geq 1$, let $c_\sigma^{[n]}$ be a function $c_\sigma^{[n]} : \Delta^{[n]} \to CH^{[n]}(\Delta)$ obtained by restricting c_σ to $\Delta^{[n]}$, where $CH^{[n]}(\Delta)$ is the set of all chain-hypergraphs with at most n nodes. Then let $\mathcal{C}^{[n]} = \{c_\sigma^{[n]} \mid c_\sigma \in \mathcal{C}\}$.

As noted in Remark 1, the alphabet Γ_Δ is infinite even if Δ is finite. This makes a trouble in discussing the PAC-learnability of a class of conformations. However, if we restrict the rank and degree of conformation rules to constant integers k and d, respectively, the alphabet $\Gamma_{(k,d,\Delta)}$ is finite for a finite alphabet Δ.

Our main result is the following theorem:

Theorem 2. Let $\mathcal{C}_{(k,d,\Delta)} = \{c_\sigma \mid \sigma \in \Gamma_{(k,d,\Delta)}^+\}$ for integers $k \geq 2$ and $d \geq 1$. Then the class $\mathcal{C}_{(k,d,\Delta)}$ is polynomial-time PAC-learnable.

6 Method of Experiments

6.1 Implementation

We have finished most of the implementation of the PAC-learning algorithm shown in the proof of Theorem 1 by Common Lisp. In our implementation, we set the rank k of bundle rule to be 4 because of the difficulty arising from time and space complexity. The algorithm CONFORM is used for predicting conformation.

6.2 Data

From PDB, we have chosen 153 proteins for our experiments. Each protein file is expressed as a distance matrix of positions of amino acid residues. In order to define the structure graph, the range ε is set to be 3Å$\sim$5Å.

The choice of the alphabet Δ for labeling the nodes of a hypergraph is a key to experiments. The alphabet Δ represents a classification of amino acid residues by their properties. In Hart and Istrail [4], they used the hydrophobic-hydrophilic model by Dill [2] that regards a protein as a linear chain amino acid residues that are of two types H (hydrophobic) and P (hydrophilic). In this case Δ is $\{H, P\}$. In addition to the alphabet Σ of amino acid residues, we will use $\Delta = \{H, P\}$ although some amino acids are neither hydrophobic nor hydrophilic.

There are many another choices for Δ by considering another properties of amino acid residues. In a final version of the paper, we will report the results of a series of experiments under various conditions and compare the results with their tertiary structures.

参考文献

[1] Berge, C., *Hypergraphs*, North-Holland, 1989.

[2] Dill, K.A., Fiebig, K.M. and Chan, H.S., Cooperatively protein-folding kinetics, *Proc. National Academy of Science, U.S.A.* **90**, 1942–1946, 1993.

[3] Harary, F., *Graph Theory*, Addison-Wesley, 1969.

[4] Hart, W.E. and Istrail, S.C., Fast protein folding in the hydrophobic-hydrophilic model within three-eights of optimal, *J. Computational Biology* **3**, No. 1, 53–96, 1996.

[5] Natarajan, B.K., Probably approximate learning of sets and functions, *SIAM J. Comput.* **20**, No. 2, 328–351, 1991.

[6] Natarajan, B.K., *Machine Learning: A Theoretical Approach*, Morgan Kaufmann, 1991.

[7] Natarajan, B.K. and Tadepalli, P., Two new frameworks for learning, Proc. Fifth International Symposium on Machine Learning, 402–415, 1988.

[8] Patton, A.L., Punch, W.F. III and Goodman, E.D., A standard GA approach to naive protein structure prediction, *Proc. Sixth International Conference of Genetic Algorithms*, Morgan Kaufmann, 574–581, 1995.

[9] Shimozono, S., Shinohara, A, Shinohara, T., Miyano, S., Kuhara, S. and Arikawa, S., Knowledge acquisition from amino acid sequences by machine learning system BONSAI, *Trans. Information Processing Society of Japan* **35**, 2009–2018, 1994.

[10] Smith, R.F. and Smith, T.F., Automatic generation of primary sequence patterns from sets of related protein sequences, *Proc. National Academy of Science* **87**, 118–122, 1990.

[11] Stolorz, P., Recursive approaches to the statistical physics of lattice proteins, *Proc. 27th Hawaii Internatinal Conference on System Sciences*, Vol. 5, IEEE Computer Society Press, 316–325, 1994.

[12] Unger, R. and Moult, J., Genetic algorithms for protein folding simulations, *J. Mol. Biol.* **231**, No. 1, 75–81, 1993.